UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/751,761	12/29/2000	Ronald D. Smith	2207/10119	5065

7590 02/27/2009

Kenyon & Kenyon
Suite 600
333 W. San Carlos Street
San Jose, CA 95110-2711

EXAMINER	
HUISMAN, DAVID J	

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
02/27/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte RONALD D. SMITH

Appeal 2008-2063
Application 09/751,761
Technology Center 2100

Decided:[1] February 27, 2009

Before HOWARD B. BLANKENSHIP, ALLEN R. MACDONALD, and JAY P. LUCAS, *Administrative Patent Judges*.

MACDONALD, *Administrative Patent Judge*.

DECISION ON APPEAL

---

[1] The two-month time period for filing an appeal or commencing a civil action, as recited in 37 CFR § 1.304, begins to run from the decided date shown on this page of the decision. The time period does not run from the Mail Date (paper delivery) or Notification Data (electronic delivery).

Appellant appeals under 35 U.S.C. § 134 from the Examiner's rejection of claims 20-38. We have jurisdiction under 35 U.S.C. § 6(b). We sustain.

STATEMENT OF THE CASE

According to Appellant, the invention relates to neutral instruction generation that can perform error checking without changing a processor system's architectural state.[2] In particular the invention involves (1) executing a neutral instruction to ascertain an architectural state value for a processor core, and (2) comparing said architectural state value for said processor core to an architectural state value for a second.[3] Claim 20 is illustrative:

20. A method comprising:

detecting a stall in an execution stage of a processor core;

generating a neutral instruction;

providing said neutral instruction to said execution state; and

executing said neutral instruction to ascertain an architectural state value for said processor core;

comparing said architectural state value for said processor core to an architectural state value for a second processor core.

The Examiner relies on the following prior art references to show unpatentability:

Ehlig	US 5,551,050	Aug. 27, 1996
Sato	US 5,903,768	May 11, 1999
Mandyam	US 6,285,974 B1	Sep. 4, 2001

[2] Spec. 2:6-8.
[3] *See* Spec. 4:18-19, 5:8-15.

Swoboda US 6,643,803 B1 Nov. 4, 2003
Hennessy and Patterson, "Computer Organization and Design, 2nd Edition, 1998.

1. The Examiner rejects claims 20, 24, 26, 33, and 37 under 35 U.S.C. § 103(a) as unpatentable over Swoboda and Ehlig (Ans. 3-7).

2. The Examiner rejects claims 21, 27, and 34 under 35 U.S.C. § 103(a) as being unpatentable over Swoboda, Ehlig, and Sato (Ans. 8).

3. The Examiner rejects claims 22, 23, 25, 28–30, 35, 36, and 38 under 35 U.S.C. § 103(a) as being unpatentable over Swoboda, Ehlig, and Mandyam (Ans. 9-11).

4. The Examiner rejects claims 31and 32 under 35 U.S.C. § 103(a) as being unpatentable over Swoboda, Ehlig, Mandyam, and Hennessy and Patterson (Ans. 11-12).

ISSUE

The issue before us is whether Appellant has shown that the Examiner erred in finding that the prior art teaches or suggests (1) executing an instruction and, if so, whether the instruction executed is a neutral instruction; (2) ascertaining an architectural state value for a processor core; and (3) comparing said architectural state value for said processor core to an architectural state value for a second processor core.

FINDINGS OF FACT

1. Independent claim 20 requires "executing said neutral instruction to ascertain an architectural state value for said processor core" (App. Br. A-1).

Claim 20 also requires "comparing said architectural state value for said processor core to an architectural state value for a second processor core" (*Id*).

2. Swoboda "relates to digital microprocessors, and more particularly to emulating and debugging digital microprocessors" (Swoboda, col. 1, ll. 21-23). The Abstract of Swoboda states that "the emulation circuitry can jam an instruction into the instruction register of the processor to cause the processor resources to be read or written on behalf of the emulation circuitry" (*See* Swoboda Abstract). When referencing Fig. 14, Swoboda discloses that "[m]emory read and write requests can be conducted by the emulation circuitry across emulation bus 853 by sending a request to processor core 102" (Swoboda, col. 22, ll. 39-41).

3. Swoboda discloses that "[t]he debug-and-test direct memory access (DT-DMA) mechanism provides access to memory, CPU registers, and memory-mapped registers (such as emulation registers and peripheral registers) without direct CPU intervention" (Swoboda, col. 26, ll. 23-26).

4. The Specification of the present application discloses that "the present invention pertains to neutral instruction generation that can perform error checking without changing the processor system's architectural state" (Spec. 2:7-8). The Specification also discloses that "an instruction which logically ORs a zero to a specified register could be a neutral instruction because nothing changes in the architectural state of the processor." (Spec. 4:14-16). Additionally, the Specification discloses that "there are many examples of neutral instructions that do not affect the architectural state of the processor" (Spec. 9:8-9).

5. A processor of Swoboda has system resources that include registers (Swoboda, col. 2, ll. 50-51).

6. The Specification of the present application discloses that architectural state values, extend to, but are not limited to, the processor registers (Spec. 5:20-22).

7. An object of Ehlig is "to aid designers in the development of emulation tools for data processing devices" (Ehlig, col. 3, ll. 10-12). Ehlig discloses an embodiment "that is useful in redundant processing applications where multiple processors are calculating the same information and voting on the answer." The embodiment "adds a compare device 18" that receives input from emulator circuit 13 and emulator circuit 14. "The compare device has an error indicate output." The "compare device continuously compares the internal data produced by data processor 200 and data processor 300." "Compare device 18 produces a signal on its error indicate output when the internal data it compares is different." (*See* Ehlig, col. 6. ll. 7-22; Fig. 3).

8. Figure 2 of Swoboda "shows the system connectivity necessary for debug with multiple CPUs in multiple devices" (Swoboda, Fig. 2; col. 5, ll. 57-59).

9. Swoboda discloses that the "[a]rchitecture and instruction set are optimized for low power consumption and high efficiency execution of DSP algorithms, such as for wireless telephones, as well as pure control tasks" (Swoboda, Abstract).

PRINCIPLES OF LAW

Claims must "particularly point out . . . the subject matter which the applicant regards as his invention." 35 U.S.C. § 112, second paragraph. "[T]he PTO gives claims their 'broadest reasonable interpretation.'" *In re Bigio*, 381 F.3d 1320, 1324 (Fed. Cir. 2004) (quoting *In re Hyatt*, 211 F.3d 1367, 1372 (Fed. Cir. 2000)). "[T]he words of a claim 'are generally given their ordinary and customary meaning.'" *Phillips v. AWH Corp.,* 415 F.3d 1303, 1312 (Fed. Cir. 2005) (en banc) (internal citations omitted).

During prosecution before the USPTO, claims are to be given their broadest reasonable interpretation, and the scope of a claim cannot be narrowed by reading disclosed limitations into the claim. *See In re Morris*, 127 F.3d 1048, 1053-54 (Fed. Cir. 1997). Also, the mere fact that there is an alternative embodiment disclosed in the Specification of the present application, does not outweigh the language of the claim. *See TIP Systems, LLC v. Phillips*, 529 F.3d 1364, 1373 (Fed. Cir. 2008).

In rejecting claims under 35 U.S.C. § 103, it is incumbent upon the Examiner to establish a factual basis to support the legal conclusion of obviousness. *See In re Fine*, 837 F.2d 1071, 1073 (Fed. Cir. 1988). In so doing, the Examiner must make the factual determinations set forth in *Graham v. John Deere Co.*, 383 U.S. 1, 17 (1966).

Discussing the question of obviousness of a patent that claims a combination of known elements, *KSR Int'l v. Teleflex, Inc.*, 127 S. Ct. 1727 (2007), explains:

> When a work is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or a different one. If a person of

> ordinary skill can implement a predictable variation, § 103 likely bars its patentability. For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill. *Sakraida v. AG Pro, Inc.,* 425 U.S. 273 (1976) and *Anderson's-Black Rock, Inc. v. Pavement Salvage Co.*, 396 U.S. 57 (1969) are illustrative—a court must ask whether the improvement is more than the predictable use of prior art elements according to their established functions.

KSR, 127 S. Ct. at 1740. If the claimed subject matter cannot be fairly characterized as involving the simple substitution of one known element for another or the mere application of a known technique to a piece of prior art ready for the improvement, a holding of obviousness can be based on a showing that "there was an apparent reason to combine the known elements in the fashion claimed." *Id.* at 1740-41. Such a showing requires

> some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness. . . . [H]owever, the analysis need not seek out precise teachings directed to the specific subject matter of the challenged claim, for a court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ.

Id. at 1741 (quoting *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006)) (internal quotation marks omitted).

In *Anderson's-Black Rock* the Court found that a combination of old elements that performs a useful function, but adds nothing to the nature and quality of the prior art is not an invention by the obvious-nonobvious standard. *See Anderson's-Black Rock* at 62-63. The Court in *KSR*, in citing

Anderson's-Black Rock, found that a combination of two pre-existing elements, which did not create a new synergy, and did no more than they would in separate, sequential operation, was obvious. *See KSR* at 1740.

If the Examiner's burden is met, the burden then shifts to the Appellant to overcome the prima facie case with argument and/or evidence. Obviousness is then determined on the basis of the evidence as a whole and the relative persuasiveness of the arguments. *See In re Oetiker*, 977 F.2d 1443, 1445 (Fed. Cir. 1992).

ANALYSIS

Claims 20, 24, 26, 33, and 37

Executing an Instruction

Claim 20 requires "executing said neutral instruction to ascertain an architectural state value for said processor core" (FF1). Appellant argues that the read instruction of Swoboda is executed by emulation circuitry that is separate from the processor. (App. Br. 5). Further, Appellant argues that "the read is being executed by a debug-and-test memory access (DT-DMA) mechanism" that is configured to operate without direct CPU intervention (Reply Br. 2). Additionally, Appellant asserts that the Examiner points to Figures 14 and 15 to show that the processor executes a neutral instruction to obtain an architectural state value. However, Appellant argues that Figures 14 and 15 teach "[e]mulation circuitry 851 provides common debug accesses … without direct CPU intervention through a [DT-DMA]" (Reply Br. 2).

The Examiner found that the abstract of Swoboda and Fig. 14 of Swoboda teach jamming an instruction into the register of the processor (Ans. 12-13). We agree with the Examiner's findings with regards to the Abstract.

The Abstract of Swoboda clearly states that "the emulation circuitry can jam an instruction into the instruction register of the processor to cause the processor resources to be read or written on behalf of the emulation circuitry" (FF 2). Further, when referencing Fig. 14, Swoboda does disclose that "[m]emory read and write requests can be conducted by the emulation circuitry across emulation bus 853 by sending a request to processor core 102" (FF 2).

The Examiner further found that "[a]n instruction register is a known processor component that stores an instruction to be executed" (underlining omitted) (Ans. 13). The Examiner then concluded that "jamming/storing an instruction into the processor's [instruction register] IR causes the processor to execute the instruction." We agree. We conclude that it would have been recognized by one of ordinary skill in the art that when the emulation circuitry sends a request to a processor core, the processor core processes, *i.e.*, executes the requests. Thus, Swoboda teaches or suggests execution of an instruction.

Swoboda does disclose that "[t]he debug-and-test direct memory access (DT-DMA) mechanism provides access to memory, CPU registers, and memory-mapped registers (such as emulation registers and peripheral registers) without direct CPU intervention," as argued by Appellant (FF 3). However, this is just one aspect of the invention in Swoboda, and does not

negate the disclosure in Swoboda of "execution of an instruction." The mere fact that there is an alternative embodiment disclosed in the Specification of the present application, does not outweigh the language of the claim (*See TIP Systems* at 1373).

A Neutral Instruction

Claim 20 requires that the instruction executed is a neutral instruction (FF 1). Appellant admits that Swoboda teaches jamming instructions into the instruction register of a processor, but argues that the Examiner has not shown that the instructions are neutral (Reply Br. 2). Appellant further argues that "the instruction being jammed might be to force the creation of a hole and halt the execution of further instructions so that the DT-DMA can perform a function" (Reply Br. 2).

The Examiner found that a "read' is a neutral instruction, as "[r]eading a resource does not modify the architectural state of the processor" (Ans. 4). The Specification of the present application does not disclose an express definition for the term "neutral." However, examples of neutral instructions referred to in the Specification of the present application are described as instructions that do not change the architectural state of a processor (FF 4).

When interpreting the term "read," we give it the broadest reasonable interpretation in light of the Specification (*See Phillips* at 1316). Thus, we agree with the Examiner, that one of ordinary skill in the art would have considered a "read" instruction to be a neutral instruction, as the "read" instruction would have only identified the contents of the processor and would not have changed the contents of the processor.

Appellant argues that an instruction being jammed might be to force the creation of a hole and to halt the execution of further instructions, so that the DT-DMA can perform a function (Reply Br. 2). While this may be so, it does not negate the disclosure, in Swoboda, of a processor that executes a neutral instruction, as discussed above. Accordingly, Swoboda discloses executing a neutral instruction.

Ascertaining an Architectural State Value for a Processor Core

Claim 20 requires "ascertaining an architectural state value for said processor core" (FF1). Appellant argues that Swoboda refers to receiving instructions, and not architectural state values of processor cores (App. Br. 6). Further, Appellant contends that the Examiner does not show that the undefined value received is the result of the execution of a neutral instruction (App. Br. 6).

The Examiner found that Swoboda's processor does receive neutral instructions (Ans. 15). The Examiner also concluded that "[i]t is the neutral instruction that, when executed, causes an architectural state value to be obtained" (Ans. 15). According to the Examiner, Swoboda discloses a neutral instruction which, when jammed into the processor, "causes a processor resource (i.e., register) to be read and the value in the register to be obtained" (Ans. 15). The Examiner concluded that the "value read from a register is an architectural state value" (Ans. 15). We agree.

As discussed above, it would have been recognized by one of ordinary skill in the art that the instruction register is a component of the processor. Thus, it would have been recognized by one of ordinary skill in the art that a reference to an instruction register being read is the same as a reference to a processor being read. Moreover, Swoboda discloses that processor resources include registers (FF 5).

The Specification of the present application discloses that architectural state values extend to, but are not limited to, the processor registers (FF 6), and Swoboda discloses that the values of the processor resources (which may include a register (FF 5)) are read (FF 2). Thus, one of ordinary skill in the art would also have recognized that a reference to reading architectural state values of a processor's register is equivalent to a reference to reading the architectural state value of a processor. Further, one of ordinary skill in the art would recognize that when Swoboda discloses reading values it is ascertaining values, as the read instruction would only obtain values from the register or processor, and not change values. Accordingly, Swoboda discloses ascertaining architectural state values for a processor.

Appellant argues that the Examiner fails to show that the undefined value received is the result of the execution of a neutral instruction (App. Br. 6). As discussed above, the "value" received is a result of the execution of a neutral instruction, as the "read" instruction is a neutral instruction. Further, we find that it is irrelevant whether the architectural state values are defined or undefined, as the language of claim 20 does not require the

architectural state values obtained to be defined or undefined. Thus, Swoboda teaches or suggests "ascertaining an architectural state value for a processor core."

Motivation to Combine Swoboda and Ehlig
Comparing First processor

Appellant argues that there is no motivation to combine Swoboda and Ehlig to meet the limitation of claim 20 that requires "comparing said architectural state value for said processor core to an architectural state value for a second processor core" (*See* App. Br. 6-8; Reply Br. 3; FF 1)

The Examiner acknowledges that "Swoboda has not taught comparing said architectural state value for said processor core to an architectural state value for a second processor core" (Ans. 4). However, the Examiner found that Ehlig teaches the concept of comparing first processor data obtained from emulation circuitry to second processor data obtained from emulation circuitry (Ans. 4).

We agree. As found in the Findings of Fact above, Ehlig discloses a compare device that continuously compares the internal data produced by data processor 200 and data processor 300, and produces a signal on its error indicate output when the internal data it compares is different (FF 7).

Appellant argues that the Examiner has not shown that the processor data in Ehlig represents an architectural state value of a processor core. The Examiner concluded that an explicit definition of "architectural state value" could not be found in the Specification (Ans. 17). The Examiner then found that "an architectural state value is simply a value produced by a processing device, and Ehlig has clearly taught such a value" (Ans. 17). Appellant's

arguments have failed to convince us of error in the Examiner's conclusion of obviousness. Further, as discussed above, we found that Swoboda teaches or suggests ascertaining architectural state values of a processor.

Appellant also argues that the invention of Swoboda is optimized for devices such as wireless telephones (App. Br. 7). Appellants contend that factors, such as "[s]ize, power consumption, cost, and many other factors make adding redundant processors to the invention of Swoboda impractical" (App. Br. 7).

The Examiner found that the Abstract states that the architecture and instruction set are optimized for high efficiency execution of algorithms for wireless telephones and pure control tasks (Ans. 20.). Based on the Findings of Fact above, we agree. The invention of Swoboda is not limited to wireless telephones, as the architecture and instruction set may be for pure control tasks.

In addition, Appellant argues that the systems of Swoboda and Ehlig would be inoperable if combined (Reply Br. 3). Appellant contends that the manual debugging method taught by Swoboda is not readily compatible with the synchronous, redundant processor system of Ehlig (*See* App. Br. 7). Appellant asserts that "manipulating a single processor would make that processor asynchronous with other processors, thus, making continuous, real-time comparisons impossible" (Reply Br. 3).

Appellant contends that the processors in Ehlig are synchronous (App. Br. 7). Further, Appellant asserts (1) that in Ehlig, the data is continuously compared during real-time operation and errors can be detected continuously (Reply. Br. 3), and (2) Swoboda 'allows multiple single processor debuggers

to be spawned' 'allow[ing] the user to manipulate each processor individually' (Reply Br. 3). Appellant also contends that, in Swoboda, (1) if one processor needs to enter a debug state, then all of the processors would be forced to do the same, and (2) forcing all of the processors to stop because of an error on one processor would defeat one of the primary purposes of having a redundant processor system (App. Br. 7).

The Examiner found that Swoboda teaches a multiprocessor system and that combining Swoboda and Ehlig would require making one of the processors a redundant processor (Ans. 20). The Examiner concluded that "redundant processors should clearly be debugged as well" (Ans. 21). The Examiner found that "[d]uring debugging of Swoboda in view of Ehlig's redundant system, the results of the cores must be compared so that it can be determined that the redundant system is working properly before deploying it" (Ans. 21). Examiner further concluded that "in order to determine that each processor is working the same on the same set of inputs, each processor must be stopped and checked against one another" (Ans. 21). The Examiner concluded that "Appellant appears to be arguing about performance when outside of the debugging environment" (Ans. 21-22). Further, the Examiner concluded that "[w]hen in the debugging /development environment, the only way to determine if each processor is performing the same at all times is to stop each processor when a result is to be checked and compare the results of each one" (App. Br. 22).

Based on the Findings of Fact above, Swoboda is directed to emulating and debugging digital multiprocessors (FF 2), and Ehlig is directed to aiding design in the development of emulation tools for data processing devices, which includes comparing data from different processors, and indicating an error if the data is different (FF 7).

Accordingly, Swoboda and Ehlig are both directed to identifying errors associated with processors. As Swoboda discloses a multiprocessor system (FF 8), it would have been obvious to make one of the processors a redundant processor, and compare the outputs of both processors, so that any processing errors would have been detected.

We find that Appellant's invention is directed to the combination of two-pre-existing elements that do no more than they would in separate, and do not create a new synergy (*See KSR* at 1740).

Therefore, for the foregoing reasons, Appellant's arguments have failed to convince us of error in the Examiner's conclusion of obviousness. Therefore, we will sustain the Examiner's rejection of independent claim 20, and dependent claim 24 that falls with claim 20. Appellant argues independent claims 26 and 33 with claim 20.[4] Independent claims 26 and 33 are commensurate in scope with independent claim 20. Thus, for similar reasons, we will sustain the Examiner's rejection of those claims, and claim 37 which falls with independent claim 33.

[4] *See* App. Br. 5-8.

OTHER REJECTIONS

We will also sustain the Examiner's obviousness rejections of (1) Claims 21, 27, and 34 over *Swoboda, Ehlig, and Sato*; and (2) claims 22, 23, 25, 28-30, 35, 36, and 38 over *Swoboda, Ehlig, and Mandyam*; and (3) claims 31 and 32 over Swoboda, Ehlig, Mandyam, and Hennessy. We find that the Examiner has reasonably reached a conclusion of obviousness that Appellant has not persuasively rebutted. Once the Examiner has satisfied the burden of supporting a conclusion of obviousness, the burden then shifts to Appellant to present evidence and/or arguments that persuasively rebut the Examiner's conclusion. *See In re Oetiker*, 977 F.2d 1443, 1445 (Fed. Cir. 1992).

Appellant has not particularly point out errors in the Examiner's reasoning to persuasively rebut the Examiner's conclusion of obviousness. Since Appellant has not persuasively rebutted the Examiner's conclusion of obviousness for claims 21-23, 25, 27-32, 34-36, and 38, the rejections of those claims are therefore sustained.

CONCLUSION OF LAW

Appellant has not shown that the Examiner erred in rejecting claims 20-38 under § 103.

DECISION

We have sustained the Examiner's rejections with respect to any of the claims on appeal. Therefore, the Examiner's decision rejecting claims 20-38 is affirmed.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. §1.136(a). See 37 C.F.R. § 1.136(a)(1)(iv).

AFFIRMED

pgc

Kenyon & Kenyon
Suite 600
333 W. San Carlos Street
San Jose, CA 95110-2711